# 基于图条件函数依赖发现的数据一致性修复算法

曹建军，余旭*（nudt_yuxu@163.com），翁年凤，袁震

(国防科技大学 第六十三研究所, 江苏 南京, 210007)

**[摘要]**以知识图谱为代表的图数据集中不可避免地会出现不完整性，冲突等数据质量问题。而数据一致性往往用以判断数据集合中的不一致或冲突，其指数据（语义）的正确性，是反映数据质量的重要指标之一。针对以数据一致性为代表的图数据的数据质量问题，基于图函数依赖（Graph Functional Dependency，GFD）提出了图条件函数依赖（Graph Conditional Functional Dependency，GCFD），并通过挖掘算法 GCFDMiner（Graph Conditional Functional Dependency Miner）和图数据一致性修复算法 GRepair 提升了图数据的数据质量。GCFDMiner 先利用频繁图挖掘算法从原图中挖掘出频繁图模式并得到同构子图，再将子图中节点与图模式相对位置进行一一映射，得到从图数据转化为关系数据的节点-属性表，最后利用关系依赖挖掘算法从表中挖掘图条件函数依赖集合。为测试图条件函数依赖的数据修复能力，通过 GRepair 迭代“匹配-修复”步骤直至成功修复不一致错误，并减弱图条件函数依赖冲突的影响提升修复效果。最后，在真实数据集和合成数据集上验证了 GCFDMiner 和 GRepair 的效果和效率优势。

**[关键词]**图条件函数依赖；数据修复；数据质量；数据一致性；图依赖；知识图谱

**[分类号]:**TP311

## An Graph Conditional Functional Dependency Discovery Based Data Consistency Repair Algorithm

Cao Jianjun, Yu Xu, Weng Nianfeng, Yuan Zhen

（The Sixty-third Research Institute, National University of Defense Technology, Nanjing Jiangsu 210007, China)

**[Abstract]**Data quality issues such as incompleteness and conflicts appear inevitably in the graph datasets represented by knowledge graphs. Data consistency is always used to determine the inconsistencies or conflicts in datasets. As an important indicators reflecting data quality, data consistency means that the correctness of data. Caused data inconsistency of graphs, we proposed graph conditional functional dependency(GCFD) based on graph functional dependency(GFD) and improved data quality of graphs by using GCFD mining algorithm GCFDMiner and GCFD repair algorithm GRepair. Firstly, GCFDMiner mines frequent graph patterns from original graph to obtain isomorphic subgraphs. Then, the vertices-attributes table from graph to relation data is matched from original graph according to the bijection of relative positions of vertices in subgraphs and graph patterns. The graph conditional dependency sets are discovered from the table by a relation dependency mining algorithm. To test the data repairing ability of GCFDs, we iterated through the matching and repairing steps by GRepair until the inconsistent errors are fixed, and improve the repairing effect by reducing the impact of graph conditional function dependency conflicts. Finally, the results on real and synthetic datasets showed the effectiveness and applicability of GCFDMiner and GRepair.

**[Keywords]**graph conditional functional dependency, data repair, data quality, data consistency, graph dependency, knowledge graph

## 1 引言

大数据时代，数据源参差不齐，不可避免地造成了数据不准确、缺失或错误，导致不同程度的数据质量问题。而低质量的数据会影响数据发现的效果，降低数据应用价值[1]。据 2002 年的报告统计，数据质量问题每年给美国商业领域业务带来 6000 亿美元的损失[2]。

根据问题发生的源头，数据质量问题可被归类为模式层问题或实例层问题[3]。为度量数据质量好坏，提出完整性、一致性、唯一性、准确性等指标[4]。其中，数据一致性（Data Consistency），指在数据集合中，每个信息都不包含语义错误或相互矛盾的数据。数据一致性作为反映数据是否真实、有效、可用的重要指标，是数据质量的重要研究内容[4-6]。

在一张图中，只有当所有子图均遵守了特定的数据质量规则（可以是预先定义或自身蕴含的）时，则该图保持了数据一致性[4]。反之，则存在数据不一致，并将违反数据质量规则的数据称为不一致数据。在图数据库中，单图中的单个子图、多个子图和多图中的多个子图都可能存在数据不一致[4]。

现有对图数据的数据质量相关研究主要集中于讨论图函数依赖（Graph Functional Dependency）[7]。图函数依赖被视为函数依赖从面向关系型数据到面向图数据的一种拓展，是最具代表性的数据质量规则之一。相比于图键（Graph Key）[8]和图关联规则（Graph Association Rule）[9]等规则，图函数依赖的语义表达能力更强，因此更具研究价值[4, 7]。

提升数据质量的一个重要方法就是针对不一致数据进行数据修复[1]。基于数据质量规则的数据修复的主要目标是通过修复规则尽可能在不引入新的错误前提下修复不一致的数据[1, 4, 10]。在关系型数据中，函数依赖将数据一致性约束在模式层。而扩展的条件函数依赖通过加入“条件属性”将约束关系具体到部分的实例子集。理论上，基于实例层的条件函数依赖更适合修复不一致的数据[11]。

因此，本文给出图条件函数依赖（Graph Conditional Functional Dependency, GCFD）的形式化定义及其依赖挖掘算法，并提出基于图条件函数依赖的数据一致性修复算法。为避免产生歧义，本文在后续内容中的图依赖包括图条件函数依赖和图函数依赖。本文的主要贡献如下：

1) 是提出图条件函数依赖的形式化定义，相比于图函数依赖，通过引入“条件属性”将约束关系具体到实例层，更适合于修复不一致数据；

2) 提出一种完整的数据一致性提升框架，包括从图数据库中挖掘得到可用的图条件函数依赖集合的图条件函数依赖挖掘算法 GCFDMiner 和基于图条件函数依赖的图数据一致性修复算法 GRepair；

3) 实验结果表明本文所提出的 GCFDMiner 算法可以有效的从图数据库中挖掘得到图条件函数依赖，而挖掘得到的图条件函数依赖在用于不一致数据修复时相比于对比数据质量规则表现出更好的查准率和召回率。

## 2 相关工作

为解决图数据的数据一致性问题，通常分为以下 2 个步骤[10-12]：1）确定数据间的潜在语义关系，即通过图依赖自动挖掘算法获得用于检测数据一致性的图依赖；2）应用图依赖检测并修复图数据集合中的不一致数据。

图函数依赖是基于面向关系型数据的函数依赖拓展而来，常见的函数依赖包括：Huhtala 等人[12]首次提出近似函数依赖，一个近似函数依赖表示为一个几乎成立的函数依赖，其并不要求所有数据实例均满足依赖关系，通过设定不同的误差度量函数与误差阈值，允许一部分数据实例不满足，从而提升了在噪声干扰条件下的鲁棒性；Bohannon 等人[13]通过添加限定条件提出了条件函数依赖，扩展了函数依赖的语义，通过限定条件缩小数据实例范围，并将约束从“模式层”扩展到“实例层”，从而提高不一致数据的捕获能力，可看作一种属性值之间存在决定关系的函数依赖。

现阶段仅少数学者研究图依赖及其自动挖掘算法[13-16, 19, 20]，但均不完善：Fan 等人[14, 15]提出了图函数依赖，并综合图模式发现算法和函数依赖发现算法构建了 GFD 生成树以尽早去除非目标图模式，尽可能减少冗余计算，再通过先纵向拓展生成图模式，再根据图模式横向拓展得到图依赖；He 等人[16]定义了 $n$-跳跃邻居图模式，以各个实体为中心，以边标签为属性将子图转化成表数据并保留图拓扑结构信息，然后基于 FP-Growth[17]算法对表格提取频繁模式，基于 CFDMiner[18]算法发现基于路径模式的图条件函数依赖；Yu 等人[19]针对 RDF 图往往包含多个属性值这一特点提出了值聚类图函数依赖，并将 RDF 图视作分解表，以 TANE 关系函数依赖算法[12]发现 RDF 中路径模式定义的函数依赖；Alipourlangouri 等人[20]为解决动态图的数据不一致问题，引入了外部的时序约束以增强图函数依赖的语义表示，提出了时序图函数依赖。

而面向图数据的一致性修复问题尚待解决，目前有关数据一致性问题的研究仅局限于关系型数据[13, 18]。相比于关系型数据，图数据有着更复杂且不规律的拓扑结构导致只有很少工作聚焦于图数据的修复工作[21-25]：Alipourlangouri 等人[20]基于关联规则挖掘算法提出了图修复规则和灵活图修复规则以达成数据修复的目的。

## 3 图条件函数依赖定义

在给出 GCFD 的形式化定义前，先给出必要的预备知识。对于图依赖挖掘，

所有图均为有向标号图，图中的每个节点和边有且仅有一个标签，标签定义了节点（边）的关键属性。

### 3.1 图依赖相关定义

定义 1 标号图（Labeled Graph）[26]。标号图指节点和边存在标签的图结构，用于表示数据内部的各种复杂关系。一个标号图记作 $G=(V, E, \Sigma, L)$ 。其中 $V$ 为图中的节点集合，其中单个节点以 $v$ 表示； $E \subseteq V \times V$ 为图中的边集合，其中单条边以 $e=(v, v')$ 表示；$\Sigma$表示图中节点标签和边标签的集合；$L$ 表示图中的标签映射函数，用来对节点和边分配相应标签，如 $L(v) \in \Sigma$ 或 $L(\mathrm{e}) \in \Sigma$ 。

定义 2 图模式（Graph Pattern）[26]。图模式用于描述图的拓扑结构，相比于图更注重于数据内部的关系而非数据本身。一个图模式记作 $Q=(V_Q, E_Q, L_Q, F_Q)$ 。其中 $V_Q$ 表示图模式中的顶点集合； $E_Q$ 表示图模式中边集合； $L_Q$ 表示图模式中顶点标签与边标签的集合； $F_Q$ 表示标签函数，用来对顶点与边分配标签， $F_Q$ 指从顶点到顶点标签，边到边标签的映射关系，即 $F_Q:(V_Q, E_Q) \rightarrow L_Q$ 。为区分图模式中不同顶点，将顶点以无实际语义的实体标签作区分，形如 $x$，$y$。如图 1(a)所示，为典型的图模式。

定义 3 图函数依赖[7]。图函数依赖以图论中图模式代表图结构，一般记作 $\varphi: Q[\mu](X \rightarrow Y)$ ，包括决定拓扑结构的图模式 $Q$，字面量集合（literals）$\mu$及决定属性约束的函数依赖 $X \rightarrow Y$ 。在图函数依赖中图模式 $Q$ 限定结构约束，依赖中的所有属性都来自于 $Q$ 匹配到的子图，并以此区分图模式中的顶点（属性）；字面量集合$\mu$包括两个（可能为空）集合 $X$ 和 $Y$，形式可以是常量字面量集合或属性字面量集合，表示为 $x.A=c$ 或 $x.A=y.B$ 。

定义 4 模式同构（Pattern Isomorphic）。模式同构问题是子图同构问题[17]的子类问题，可被理解为不考虑顶点标签的子图同构，即当数据图中存在子图，与给定模式图在拓扑结构和边标签完全匹配，则被认定为模式同构，其具体定义如下。给定图 $G=(V, E, \Sigma, L)$ 与图模式 $Q=(V_Q, E_Q, \Sigma_Q, L_Q)$ ，若使 $G$ 与 $Q$ 模式同构，只需存在一个双射函数满足以下条件：

1） $\forall u, v \in V, (u, v) \in E \Rightarrow (f(u), f(v)) \in E_Q$ ；

2） $\forall (u, v) \in E, L((u, v))=L_Q((f(u), f(v)))$ 。

图 1 展示了基于模式同构的图模式匹配过程。

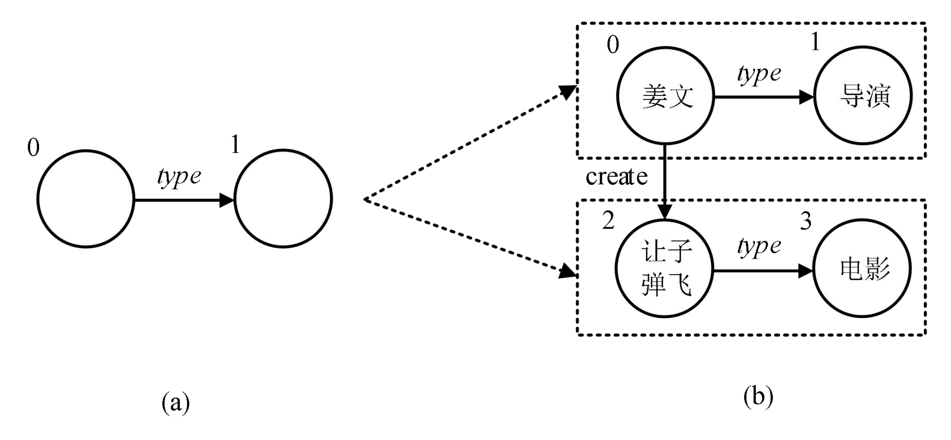



图 1　图模式匹配示例 (a)模式图示例 (b)数据图示例

Fig. 1　An example of graph pattern matching. (a)A pattern graph (b)A data graph

定义 5 图条件函数依赖。一个图条件函数依赖记作 $\varphi: Q[\mu](X \to Y,\ t_p[X \cup Y])$ 。相比于图函数依赖，图条件函数依赖增加了常量值 $t_p[X \cup Y]$ ，其为节点在值域范围 $X \cup Y$ 内某个具体的取值，即对于 $a \in X \cup Y$ ， $t_p[a]$ 为变量值或 $a$ 的值域中某个特定的常量值。

3.2　数据修复相关定义

给定图依赖集合 $\Sigma_{GFD}$ ，如果图数据库中存在子图 $G$ 违背了 $\Sigma_{GFD}$ 中任意一个图依赖，则称 $G$ 相对于 $\Sigma_{GFD}$ 不一致。本文所研究的数据修复是指对于数据集合的修改，而非添加或删除数据，并将修复目标设定为集合最小化修复。

定义 6 集合最小化修复（Set-minimal Repair）[11]。令修复集合 $\Delta(G, G')$ 表示在数据修复中发生变更的所有子图的节点修改集合，当且仅当不存在被修复图 $G''$，使得 $\Delta(G, G'') \subset \Delta(G, G')$ ，并且对于任意节点 $v \in \Delta(G, G'')$ ， $G''(v) = G'(v)$ ，称一个对图数据库子图 $G$ 的修复图 $G'$是集合最小化修复。

定义 7 图修复问题[11]。给定图 $G$，图依赖集合 $\Sigma_{GFD}$ 和真值集合 $\Sigma_T$ ，得到修复集合 $\Delta(G, G')$ 和被修复的图 $G'$，并使得 $G'$满足集合最小化修复。

考虑到在数据一致性领域尚无公认的标准数据集[1]。并且难以确保在利用图依赖进行一致性检测与修复时，其图依赖左部数据全部源于真值集合。因此，当利用图依赖进行一致性修复时，默认图依赖左部数据全部正确。

## 4 算法

本节详细介绍面向图数据的数据一致性提升框架，包括图条件函数依赖挖掘算法 GCFDMiner 和基于图条件函数依赖的数据一致性修复算法 GRepair。首先介绍两个算法的总体框架，然后依次介绍框架中的各个重要算法。GCFDMiner 框架可被形式化描述为：给定图 $G$ 和频繁度阈值$\sigma$、$k$，得到满足阈值的图条件函数依赖集合；GRepair 框架可被形式化描述为：给定被噪声污染的图 $G_n$ 和图条

件函数依赖集合 $\Sigma_{GCFD}$，得到被修复图 $G'$。具体而言，数据一致性提升框架如图 2 所示。

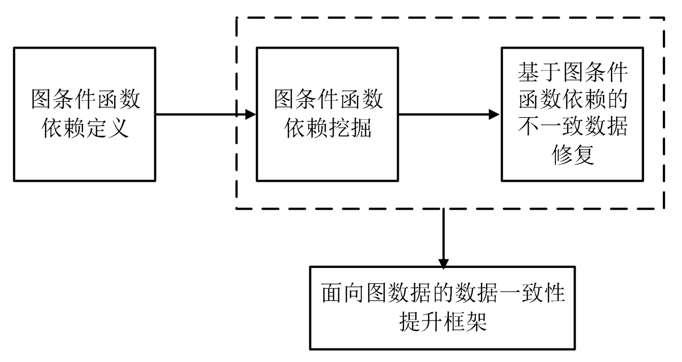



图 2　数据一致性提升框架

Fig. 2　A framework of data consistency improvement

4.1　图条件函数依赖挖掘算法 GCFDMiner

给定一个图 $G$ 和相关阈值$\sigma$、$k$，先后通过频繁图挖掘算法得到$\sigma$-频繁图模式 $\Sigma_{GP}$，图匹配算法得到顶点-属性映射表 $T_q$ 和 $k$-频繁条件函数依赖发现算法得到($\sigma$, $k$)-GCFD 集合 $\Sigma_{GCFD}$。具体过程如算法 1 所示。

**算法 1**　GCFDMiner

输入：图 $G$，频繁度阈值$\sigma$，$k$。

输出：满足阈值的$\sigma$-GCFD 集合 $\Sigma_{GCFD}$。

a) $\Sigma_{GP}$ = $GraphMining(G, \sigma)$。

b) for $q$ in do $\Sigma_{GCFD}$ :

c) 　$T_q$ = $GraphMatching(G, q)$。

d) 　$\Sigma_{GCFD}$ = $\Sigma_{GCFD} \cup CFDMining(T_q, k)$。

e) 返回 $\Sigma_{GCFD}$。

算法 1 主要包含 3 个阶段，分别为第 a 行图挖掘阶段 GraphMining( )、第 c 行的图匹配阶段 GraphMatching( )、第 d 行的依赖挖掘阶段 CFDMining( )。算法流程图及各阶段对应输入输出如图 3 所示。

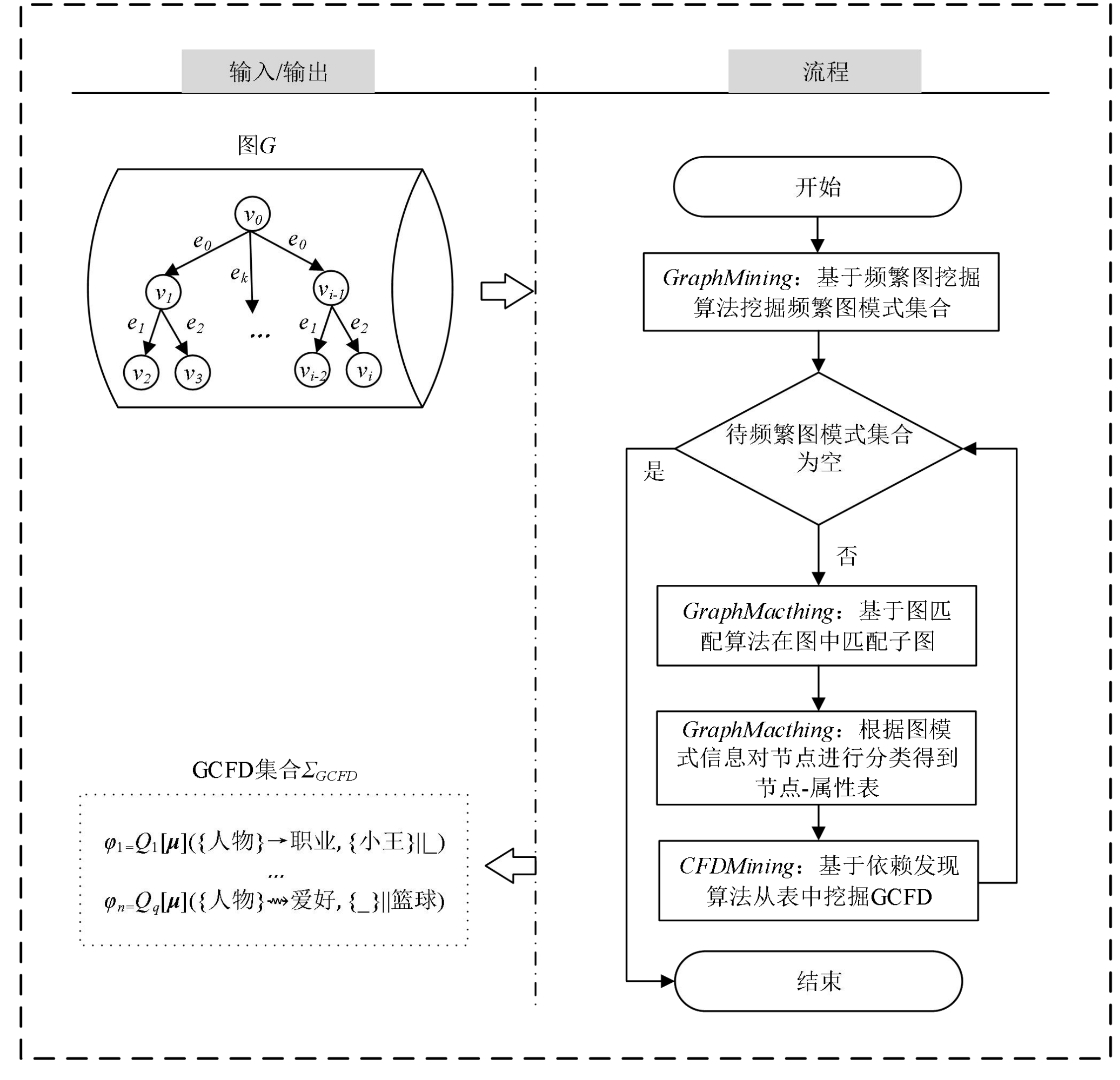



图 3　GCFDMiner 流程图

Fig. 3　A framework of GCFDMiner

在图 3 中，$v_1$~$v_i$ 表示节点标签，每个节点标签在图 $G$ 中唯一，为常量，如"姜文"；$e_1$~$e_k$ 表示边标签，每个边标签在图 $G$ 中不唯一，为常量，如"职业为"；$x$（$y$）表示图模式中变量，用于区分不同节点及在图模式匹配过程中映射到对应节点。

GCFDMiner 将图条件函数依赖挖掘任务进行划分。通过图-表转化策略将图条件依赖挖掘任务划分为三个子任务：图挖掘阶段、图匹配阶段和依赖挖掘阶段。在依赖挖掘阶段中，目标为从关系型数据中挖掘条件函数依赖。而从关系型数据库中挖掘近似函数依赖已有了相当丰富的研究。因此，在依赖挖掘阶段引入 TANE 算法[12]。GRAMI 算法[27]和 TANE 算法[12]不是本文主要工作，不做赘述。

### 4.2　图匹配阶段

由于基于搜索的匹配技术无法满足大数据高效处理的需求，现有图匹配技术多采用基于索引的思想预先选择数据图中有效特征建立倒排索引，从而减少搜索空间[25, 27]。由于本文所研究的图模式不带有节点标签，因此，将图模式看作边集合。

基于索引的图匹配技术的基本思想主要包括索引建立和模式匹配两部分。在利用边标签建立索引后，引入缩减的 DFS 编码[27]减少模式匹配时间。DFS 编码

自提出始便用来解决子图同构问题，即两个图同构当且仅当它们的最小 DFS 编码相同。根据所研究的图模式特点，将五元组 DFS 编码(节点 1，节点 2，节点 1 标签，边标签，节点 2 标签)缩减为三元组 DFS 编码(节点 1，节点 2，边标签)从而进一步减少匹配时间。图匹配阶段具体过程如算法 2 所示。

算法 2　GraphMatching

输入：图 $G$，图模式 $q$。

输出：节点-属性映射表 $\tau_q$。

a) $E_q \leftarrow$提取图 $G$ 中所有边的边序列。

b) $C_q \leftarrow$生成图模式 $q$ 对应的 DFS 编码序列集合 $C$。

c) for 边标签 $l$ in $E_q$：

d) 　$I_l \leftarrow$ 在 $G$ 中提取所有标签为 $l$ 的边的索引。

e) 　图模式中边对应 DFS 编码 $c \leftarrow C_q.\text{pop}()$。

f) 　标签 $l \leftarrow$ 提取 $c$ 中边标签。

g) 　匹配图索引集合 $I_g \leftarrow I_l$。

h) 　while $C_q \neq \varnothing$：

i) 　　$c \leftarrow C_q.\text{pop}()$。

j) 　　$C_s \leftarrow C_s \cup c$。

k) 　　$l \leftarrow$ 提取 $c$ 中边标签。

l) 　　for 索引 $i$ in $I_g$：

m) 　　　for 索引 $i'$ in $I_l$：

n) 　　　　if 根据 $i$，$i'$生成的子图满足 $C_s$：

o) 　　　　　$I_g \leftarrow i \cup i'$。

p) $\tau_q \leftarrow$根据 $I_g$ 在 $G$ 中得到相应边的顶点标签。

q) 返回 $\tau_q$。

其中，第 a 行通过获取边索引可以有效缩小检索空间；第 d 行则是顺序取出单条 DFS 编码从而找到对应边进行匹配；第 h~o 行的代码为在现有图的基础上，仅添加一条边生成新的子图。

### 3.3　数据一致性修复算法 GRepair

给定被污染的图 $G_n$，图条件函数依赖集合 $\Sigma_{GCFD}$，得到被修复的图 $G'$。具体过程如算法 3 所示。

算法 3　GReair

输入：图 $G_n$，图条件函数依赖集合 $\Sigma_{GCFD}$。

输出：被修复图 $G'$。

a) $\Sigma_{GP}, \Sigma_{CFD}$ = Separate($\Sigma_{GCFD}$)。　// 拆分图条件函数依赖集合

b) while $G'$ 更新 do: // $G'$不再更新说明迭代已经收敛

c) for 图模式 $q$ in $G'$ do:

d) $T = GraphMatching(G, q)$。

e) for $CFD$ in $\Sigma_{CFD}$ do:

f) $\Delta(G, G') = \Delta(G, G') \cup ErrorRepairing(T, CFD)$。 // 利用 $CFD$ 修复错误并保存修复记录$\Delta(G, G')$。

g) $G' = G' \cup \Delta(G, G')$。 // 对图进行修复

h) 返回 $G'$。

其中，第 a 行的 Separate( )将图函数依赖集拆分图模式集合和各图模式对应的条件函数依赖集合；第 b 行的通过迭代修复直至数据的全部错误被修复；第 f 行的 ErrorRepairing( )为错误修复算法，其并不直接对图进行修改，而是先保存修复记录，汇总后再进行修复从而避免修复记录冲突。

## 4 实验

### 4.1 实验环境

实验在基于 Ubuntu16.04 的服务器实现，CPU 型号为 AMD Ryzen 7 5800H，主存为 32GB，算法基于 Python3.11 实现。

### 4.2 数据集

实验选取了 2 个 RDF 数据集：FB15k-237[28]和 WN18RR[29]。其中 FB15k-237 包含 14951 个实体，1345 种关系和 50000 条三元组事实；WN18RR 包含 40943 个实体，11 种关系和 86835 条三元组事实。为测试修复算法有效性，针对特定图依赖$\varphi: Q[\bar{x}](X \to Y)$，将节点-属性映射表中 $a$%的记录（元组）的属性 $Y$ 的值随机改变。其中，在节点-属性映射表中，每一条代表一个图模式，而表中的属性对应图模式的节点。

实验评估被设计来考察 4 个关键问题：GCFDMiner 算法有效性、GRepair 算法有效性和图条件函数依赖的优越性、GRepair 算法中 GCFD 数量的影响。并分别考虑了不同的参数设置对算法效率的影响，包括噪声率 $a$ 和频繁度$\sigma$的数值设置、迭代策略的高效性。

### 4.3 对比方法

在验证 GCFD 的优越性与有效性的实验中，设置了其他数据质量规则进行对比。以下是对比方法的简要介绍：

图函数依赖（GFD）[7]。由函数依赖与图模式组成，指指定图模式的节点间存在语义约束关系，将一致性约束在模式层。

图关联规则（GAR）[30]。由关联规则与图模式组成，指节点所在图模式决定节点间的关联关系，将一致性约束在实例层。

4.4 评价指标

采用召回率（Recall），查准率（Precision）和 $F1$ 值作为评价指标衡量算法的检测效率，各指标的计算公式分别如式(1)，式(2)，式(3)所示。

$$Recall = \frac{|V^{GD} \cap V^{E}|}{|V^{E}|} \qquad (1)$$

$$Precison = \frac{|V^{GD} \cap V^{E}|}{|V^{GD}|} \qquad (2)$$

$$F1 = \frac{2 \times \text{Recall} \times \text{Percision}}{\text{Recall} + \text{Percision}} \qquad (3)$$

其中，$V^{E}$ 代表引入的噪声，即代表实际违反图条件函数依赖的节点；$V^{GD}$ 代表被图条件函数依赖所捕获并判断为噪声的节点。

4.5 实验结果

实验评估所考察的关键问题如下：

a) GCFDMiner 算法的有效性。

在不同的数据集以及不同的参数设置条件下，验证了实验 a 的效果，实验结果如表 1 所示。

表 1　GCFDMiner 挖掘结果

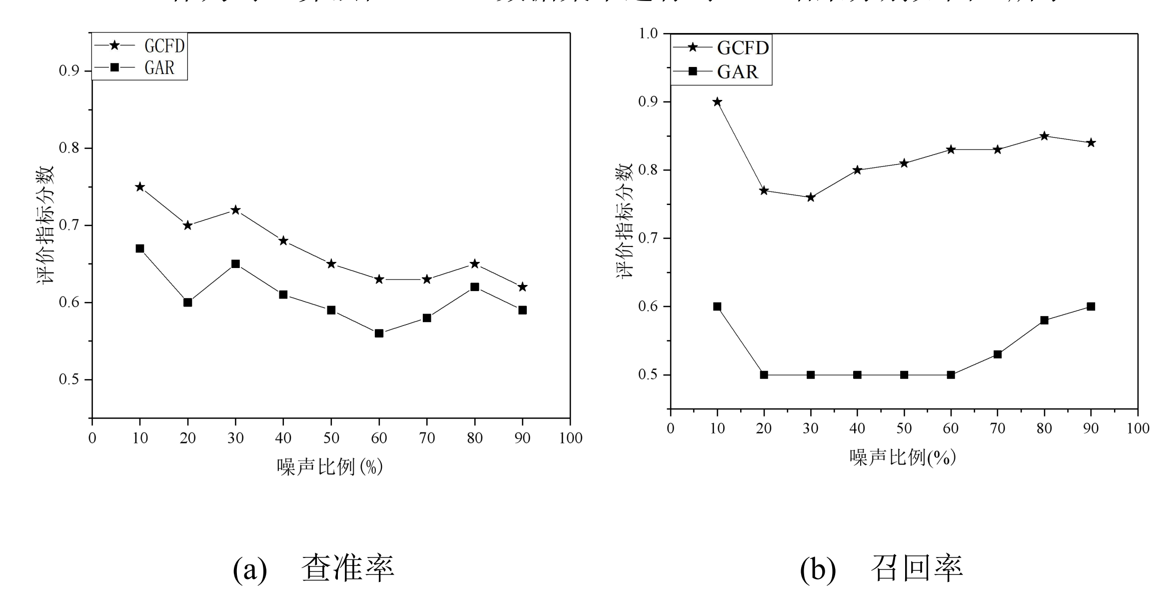
Table 1　Results of GCFDMiner

| Datasets | Frequency | | Number |
|---|---|---|---|
| | $\sigma$ | $k$ | |
| FB15k-237[28] | 250 | 250 | 149 |
| FB15k-237[28] | 250 | 500 | 47 |
| FB15k-237[28] | 250 | 750 | 23 |
| FB15k-237[28] | 500 | 250 | 45 |
| FB15k-237[28] | 500 | 250 | 22 |
| FB15k-237[28] | 500 | 250 | 13 |
| WN18RR[29] | 100 | 100 | 53 |
| WN18RR[29] | 100 | 150 | 43 |
| WN18RR[29] | 100 | 200 | 12 |
| WN18RR[29] | 200 | 100 | 31 |
| WN18RR[29] | 200 | 150 | 11 |
| WN18RR[29] | 200 | 200 | 3 |

表 1 描述了 GCFDMiner 的挖掘结果。其中，频繁度$\sigma$为图模式的频繁度，频繁度 $k$ 为条件函数依赖的频繁度。从表 1 中可以看出，GCFD 的数量受频繁度 $k$ 的影响更大。且当频繁度$\sigma$或频繁度 $k$ 减少时，GCFD 的数量均上升。

b) GRepair 算法有效性和 GCFD 的优越性。

为验证在不一致修复过程中图条件函数依赖的优越性，将图模式关联规则（GAR）作为对比算法在 FB15k 数据集中进行对比，结果分别如图 4 所示。




(a) 查准率 (b) 召回率

图 4 FB15k 数据集中数据一致性修复结果

Fig. 4 Results of data consistency repair on FB15k

由图可知，在绝大多数情况下，GCFD 有着更高的召回率和查准率，相比于 GAR，将查准率从 0.61 提升到了 0.XX，将召回率从 0.53 提升到了 0.XX。当噪声比例上升时，GCFD 和 GAR 的查准率呈下降趋势，召回率呈先下降后上升趋势，这可能是因为 GCFD 和 GAR 集合中低质量的规则太多，没有做好筛选。

c) GRepair 算法中 GCFD 数量来的影响。

为验证在不一致修复过程中 GRepair 算法的表现是否稳定，通过控制 GCFD 数量来实现不同环境。在 FB15k 数据集上进行了实验 c，结果如图 5 所示。

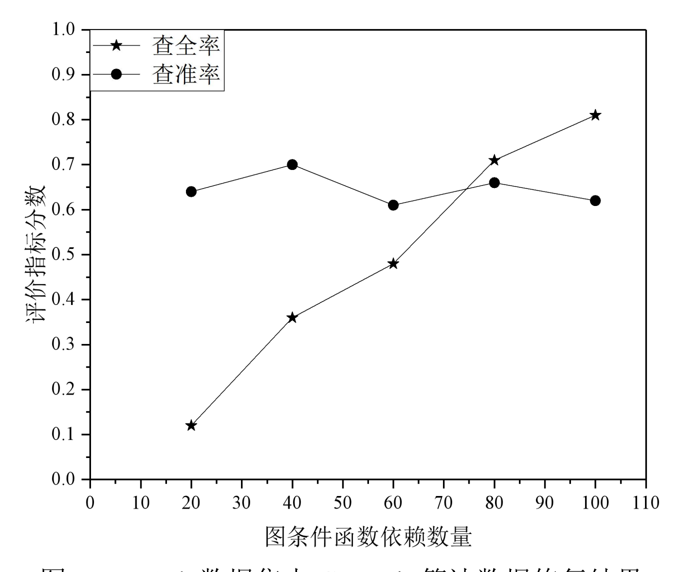



图 5 FB15k 数据集中 GRepair 算法数据修复结果

Fig. 5 Results of performance of GRepair on FB15k

由图 5 可知，在绝大多数情况下，GCFD 有着更高的查准率和召回率，相比于 GAR，将查准率从 0.61 提升到了 0.XX，将召回率从 0.53 提升到了 0.XX。当噪声比例上升时，GCFD 和 GAR 的查准率呈下降趋势，召回率呈先下降后上升趋势，这可能是因为 GCFD 和 GAR 集合中低质量的规则太多，没有做好筛选。

d）迭代修复的高效性

为测试迭代修复在不一致修复过程中的高效性，选取了 XX 个图条件函数依赖，将噪声比例设为 XX%，测试结果如图 5-5 所示。

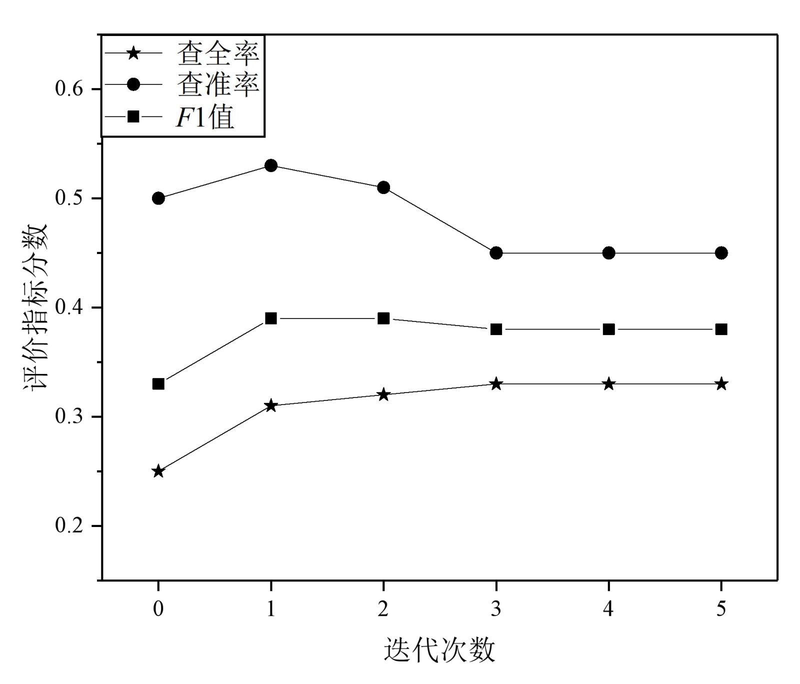



图 6　GRepair 算法数据修复结果

Fig. 6　Results of performance of GRepair

由图 6 可知，随着迭代次数的增加，查准率下降，查全率上升，$F1$ 值先上升后下降，最后均趋于稳定。仅从数据修复的角度出发，单纯的增加迭代次数反而会使查准率下降，无法达成目的；若从提高数据一致性的角度出发，$F1$ 和查全率的上升趋势说明了在一定程度下，迭代修复可以发现更多的错误，只是需要提前停止迭代。

查准率的下降最有可能是因为上一轮的错误修复造成原本不会被图条件函数依赖捕获的图被捕获，从而对数据集产生污染。

## 5　结论

本文为提升图数据库的数据一致性，提出图函数依赖的拓展——图条件函数依赖 GCFD，并提出了完整的数据一致性提升框架，包括图条件依赖挖掘算法 GCFDMiner 以及图数据一致性修复算法 GRepair，在数据集上表现良好。主要贡献如下：

1) 给出了 GCFD 的形式化定义，相比于图函数依赖，通过引入“条件属性”将约束关系具体到实例层，更适合于修复不一致数据，相比于图关联规则表现出更好的查准率和召回率；

2) 提出一种完整的数据一致性提升框架，包括图条件函数依赖挖掘算法 GCFDMiner 和基于图条件函数依赖的图数据一致性修复算法 GRepair。

未来可以在以下几个方面进行更深一步的研究和完善：

1) 论文在图函数依赖的基础上提出了图条件函数依赖，受限于依赖自身的语义表达能力，对数据一致性的提升有限，下一步可以提出语义表达更为复杂的图函数依赖的拓展，并进行深入研究。

2) 论文所研究的不一致修复方法仅针对错误属性值的修复。然而，在图数据库中，常见的数据修复还包括缺失关系的预测以及实体分辨等问题，而如何构建基于图函数依赖的关系预测或实体分辨模型值得进一步深入研究。

3) 在工业界应用中，获取图函数依赖是最重要的一步。但图函数依赖自动挖掘算法往往耗时巨大，需要更深一步的优化。因此，为使图函数依赖相关应用真正落地，设计相应优化算法是重中之重。

**参考文献：**

[1]Fan W F. Dependencies for graphs: challenges and opportunities[J]. ACM Journal of Data and Information Quality, 2019, 1(1): 1-10.

[2]Eckerson W W. Data quality and the bottom line: achieving business success through a commitment to high quality data[R]. Seattle: Technical Report, 2002

[3]Rahm E., Do H H. Data cleaning: problems and current approaches[J]. IEEE Data Engineering Bulletin, 2000, 23(4):3~13.

[4]余旭, 曹建军, 翁年凤, 等. 图依赖研究与应用综述[J]. 计算机应用研究, 2023, 40(5): 1312-1317. (Yu X, Cao J J, Weng N F, et al. Survey on research and application of graph dependencies[J]. Application Research of Computers, 2023, 40(5): 1312-1317.)

[5]Fan W F, Geerts F, Tang N, et al. Conflict resolution with data currency and consistency[J]. ACM Journal of Data and Information Quality, 2014, 5(12).

[6]Fan W F, Geerts F, Ma S, et al. Data quality problems beyond consistency and deduplication[M] // In Search of Elegance in the Theory and Practice of Computaion. Springer Berlin Heidelber, 2013: 237-249.

[7]Fan W F, Wu Y H, Xu J B. Functional dependencies for graphs[C]// Proc of the 2016 International Conference on Management of Data. New York: ACM, 2016: 1843-1857.

[8]Fan W F, Fan Z, Tian C, et al. Keys for graphs[J]. Proc of the VLDB Endowment, 2015, 8(12): 1590-1601.

[9]Fan W F, Wang X, Wu Y H, et al. Association rules with graph patterns[J]. Proc of the VLDB Endowment, 2015, 8(12): 1502-1513.

[10]Cheng Y, Chen L, Yuan Y , et al. Strict and flexible rule-based graph repairing[J]. IEEE Trans on Knowledge and Data Engineering, 2022, 34(7): 3521-3535.

[11]金澈清, 刘辉平, 周傲英. 基于函数依赖与条件约束的数据修复方法[J]. 软件学报, 2016, 27(7): 1671−1684.(Jin C Q, Liu H P, Zhou A Y. Functional dependency and conditional constraint based data repair[J]. Journal of Software, 2016, 27(7): 1671-1684.)

[12]Huhtala Y, Kärkkäinen J, Porkka P, et al. TANE: an efficient algorithm for discovering functional and approximate dependencies[J]. Computer Journal, 1999, 42(2): 100-111.
[13]Bohannon P, Fan Wenfei, Geerts F, et al. Conditional functional dependencies for data cleaning[C]∥Proceedings of the 23rd International Conference on Data Engineering. Piscataway: IEEE, 2007: 746-755.
[14]Fan Wenfei, Hu Chunming, Liu Xueli, et al. Discovering graph functional dependencies[C]∥Proceedings of the 2018 International Conference on Management of Data. New York: ACM, 2018: 427-439
[15]Fan F, Liu L, Lu P, et al. Catching numeric inconsistencies in graphs[J]. ACM Trans on Database Systems, 2020, 45(2): 1-47.
[16]He B B, Zou L, Zhao D Y. Using conditional functional dependency to discover abnormal data in RDF graphs[C]// Proc of Semantic Web Information Management. New York: ACM, 2014: 1-7.
[17]Han J, Pei J, Yin Y, et al. Mining frequent patterns without candidate generation: a frequent-pattern tree approach[C]// Data Mining and Knowledge Discovery. New York: ACM, 2004: 53-87.
[18]Fan W F, Geerts F , Li J Z, et al. Discovering conditional functional dependencies[J]. IEEE Trans on Knowledge and Data Engineering, 2011, 23(5): 683-698.
[19]Yu Y, Heflin J. Extending functional dependency to detect abnormal data in RDF graphs[C]∥ Proceedings of the 10th International Conference on the Semantic Web. Berlin: Spring, 2011: 794-809.
[20]Alipourlangouri M, Fei Chiang A M, Wu Y H. Temporal graph functional dependencies[J]. Proc of the VLDB Endowment, 2020, 14(1). arXiv: 2108.08719v2.
[21]Taleb I, Serhani M A, Bouhaddioui C, et al. Big data quality framework: a holistic approach to continuous quality management[J]. Journal of Big Data, 2021, 8(1): 1-41.
[22]Xiao R J, Yuan Y A, Tan Z J, et al. Dynamic functional dependency discovery with dynamic hitting set enumeration[C]// The 2022 IEEE 38th IEEE International Conference on Data Engineering. Piscataway: IEEE, 2022: 286-298.
[23]Qahtan A, Tang Nan, Ouzzani M, et al. Pattern functional dependencies for data cleaning[J]. Proc of the VLDB Endowment, 2020, 31(5): 684-697.
[24]Xiao R J, Yuan Y A, Tan Z J, et al. Dynamic functional dependency discovery with dynamic hitting set enumeration[C]// The 2022 IEEE 38th IEEE International Conference on Data Engineering. Piscataway: IEEE, 2022: 286-298.
[25]Ma J T, Wang Y J, Chen X T, et al. NGDcrm: a numeric graph dependency-based conflict resolution method for knowledge graph[J]. High Technology Letters, 2021, 27(2): 153-162.
[26]刘勇. 图模式挖掘技术的研究[D]. 哈尔滨: 哈尔滨工业大学, 2010.(Liu Y. Research on Techniques for Mining Graph Patterns[D]. Haerbin: Haerbin Institute of Technology, 2010.)
[27]Elseidy M, Abdelhamid E, Skiadopoulos S, et al. GRAMI: Frequent subgraph and pattern mining in a single large graph[J]. Proceedings of the VLDB Endowment, 2014, 7(7): 517-528.
[28]Bollacher K, Evans C, Paritosh P, et al. Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge[C]∥ Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. New York: ACM, 2008: 1247-1250.
[29]Dettmers T, Pasquale M, Pontus S, et al. Convolutional 2D Knowledge Graph Embeddings[C]∥ Proceedings of the AAAI Conference on Artificial Intelligence. Palo Alto: AAAI, 2018, 32(1): 1811-1818.
[30]Fan W F, Wang X,Wu Y H, et al. Association Rules with Graph Patterns[J]. Proceedings of the VLDB Endowment, 2015, 8(12): 1502-1513.